

# METHOD FOR CREATING A BALANCED BINARY TREE

## TECHNICAL FIELD

This invention relates generally to the creation of binary trees, and, more particularly,  
5 to the creation of balanced binary trees.

## BACKGROUND OF THE INVENTION

0 0976401.1.01.001  
1 A binary tree is a data structure having a series of linked nodes arranged in a tree-like  
2 pattern. The top node is often referred to as the "root." Each node in the tree has a least one  
3 and as many as two pointers to child nodes. A "balanced" binary tree is generally  
4 symmetrical from left to right and has the same or nearly the same number of elements on  
5 the left and right sides. Binary trees are used in many types of computer programs. For  
6 example, in database programs, binary trees are used to facilitate searching for database  
7 records. Each node might represent a range of index numbers and, instead of sorting through  
8 the index numbers from start to finish, which can be time consuming, the database program  
9 can simply walk through the tree along nodes that are in the correct range until it reaches a  
10 node that points to the sought-after record.

11 Binary trees work best when they are balanced. Once an unbalanced binary tree has  
12 been created, it is time consuming and memory intensive to rebalance the tree. Data that is  
13 to be assembled into a tree can be pre-processed to insure that, once the tree is built, that it is  
14 balanced. This is also time consuming, however. Thus it can be seen that there is a need for  
15 an improved method for creating a balanced binary tree.

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

1284

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

1309

1310

1311

1312

1313

1314

1315

1316

1317

1318

1319

1320

1321

1322

1323

1324

1325

1326

1327

1328

1329

1330

1331

1332

1333

1334

1335

1336

1337

1338

1339

1340

1341

1342

1343

1344

1345

1346

1347

1348

1349

1350

1351

1352

1353

1354

1355

1356

1357

1358

1359

1360

1361

1362

1363

1364

1365

1366

1367

1368

1369

1370

1371

1372

1373

1374

1375

1376

1377

1378

1379

1380

1381

1382

1383

1384

1385

1386

1387

1388

1389

1390

1391

1392

1393

1394

1395

1396

1397

1398

1399

1400

1401

1402

1403

1404

1405

1406

1407

1408

1409

1410

1411

1412

1413

1414

1415

1416

1417

1418

1419

1420

1421

1422

1423

1424

1425

1426

1427

1428

1429

1430

1431

1432

1433

1434

1435

1436

1437

1438

1439

1440

1441

1442

1443

1444

1445

1446

1447

1448</

## SUMMARY OF THE INVENTION

In accordance with this need, a computer implemented method for creating a balanced binary tree is provided, in which a list of elements is successively divided into left side and right side groupings. The median of each left side grouping is placed in the tree as a  
5    descendent node of the previous median until the list can no longer be divided. The method then involves walking back up the tree to find right-side medians and inserting them as right side descendent nodes into the binary tree. Whenever there is a choice between breaking down a left side grouping and breaking down a right side grouping, the left side grouping takes priority. The method may also be implemented so that right side groupings take  
10    priority.

Additional features and advantages of the invention will be made apparent from the following detailed description of illustrative embodiments that proceeds with reference to the accompanying figures.

## BRIEF DESCRIPTION OF THE DRAWINGS

While the appended claims set forth the features of the present invention with particularity, the invention, together with its objects and advantages, may be best understood from the following detailed description taken in conjunction with the accompanying drawings of which:

- 20        FIG. 1 is an example of a computer network;  
         FIG. 2 is an example of a computer;  
         FIGS. 3-4 shows the general setup of an embodiment of the invention;

FIG. 5 is a flowchart showing an example of steps that may be followed to create a balanced binary tree according to an embodiment of the invention;

FIGS. 6-11 show how an embodiment of the invention progresses through a list to create a binary tree;

5 FIG. 12 is a flowchart showing an example of a recursive implementation of an embodiment of the invention; and,

FIG. 13 shows the general setup of a more specific embodiment of the invention.

### **DETAILED DESCRIPTION OF THE INVENTION**

10 The invention is generally directed a method for balancing a binary tree in which a computer or equivalent device receives a sorted linked list as input. The computer then builds the left side child nodes of the binary tree by successively subdividing the list and finding the median element of each subdivision. Each successive median is then linked to the previous median as a left child node in the binary tree. The computer then repeats this  
15 procedure for the right side subdivisions, working its way back up the tree to build the right side child nodes. Of course, it is understood that the procedure may be modified so that the right-side child node are built first and so that priority is given to the right side subdivisions.

Although it is not required, the invention may be implemented by computer-executable instructions, such as program modules, that are executed by a computer.

20 Generally, program modules include routines, programs, objects, components, data structures and the like that perform particular tasks or implement particular abstract data types. On many computers, modules execute within an address space of the computer's memory, which is typically defined as a "process." The point of execution of the program instructions is

often referred to as a "thread." As is conventional, multiple threads of execution may exist for a single program in a process. Multiple processes may be executed on a single machine, with each process having one or more threads of execution. Thus, when multiple threads are discussed herein, it may mean multiple threads in a single process or multiple threads in  
5 different processes.

The invention may be implemented on a variety of types of computers, including personal computers (PCs), hand-held devices, multi-processor systems, microprocessor-based on programmable consumer electronics, network PCs, minicomputers, mainframe computers and the like. The invention may also be employed in distributed computing environments, where tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, modules may be located in both local and remote memory storage devices.  
10

An example of a networked environment in which this system may be used will now be described with reference to FIG. 1. The example network 102 includes several computers 100 communicating with one another. The network 102 may include many well-known components, such as routers, gateways, hubs, etc. and may allow the computers 100 to communicate via wired and/or wireless media.  
15

Referring to FIG. 2, an example of a computer on which the invention described herein may be implemented is shown. In its most basic configuration, the computer, generally labeled 100, typically includes at least one processing unit 112 and memory 114. Depending on the exact configuration and type of the computer, the memory 114 may be volatile (such as RAM), non-volatile (such as ROM or flash memory) or some combination of the two. This most basic configuration is illustrated in FIG. 2 by dashed line 106. The  
20

computer 100 may also have additional features/functionality. For example, computer 100 may include additional storage (removable and/or non-removable) including, but not limited to, magnetic or optical disks or tape. Computer storage media includes volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage of information such as computer readable instructions, data structures, program modules, or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disk (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by the computer 100. Any such computer storage media may be part of computer 100.

Computer 100 may also contain communications connections that allow it to communicate with other devices. A communication connection is an example of a communication medium. Communication media typically embodies computer readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. By way of example, and not of limitation, communication media includes wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, RF, infrared and other wireless media. The term computer-readable media as used herein includes both storage media and communication media.

Computer 100 may also have input devices such as a keyboard, mouse, pen, voice input device, touch input device, etc. Output devices such as a display 118, speakers, a printer, etc. may also be included. All these devices are well known in the art and need not

be discussed at length here.

Referring to FIGS. 3 and 4, an embodiment of the invention will now be described. In this embodiment, it is assumed a server computer 200 (FIG. 3) is communicatively linked to a client computer 202. A set 204 of ten records (labeled 1a-10a) is stored on the server computer 200. It is understood that the number of records shown is an example, and that the invention may work for any number of records. Each record of the set 204 has a respective index value from 1 to 10. The records are initially unsorted. The client computer 202 retrieves the records and insertion-sorts them into a data structure 206 that is a combination of a linked list 212 and an unassembled binary tree 220. As shown more clearly in FIG. 4, the linked list 212 includes a series of objects 1b-10b. The unassembled tree 220 itself includes nodes 1c-10c that are eventually to be assembled. Each object 1b-10b of the list 212 represents one of the records 1a-10a of the set 204 (from FIG. 3). Each object 1b-10b includes a pointer 214 to the next object in the list, a pointer 216 to one of the nodes 1c-10c, and a Boolean flag 218, initially set to FALSE, to indicate whether the record represented by the object has been placed in its final location as a node in the assembled binary tree.

Each of the nodes 1c-10c of the unassembled tree 220 includes a left child pointer 224 that points to the node's left child, a right child pointer 226 that points to the node's right child, and a pointer 228 to the record itself. As can be seen from FIGS. 3 and 4, the linked list 212 acts as a wrapper around the nodes of the unassembled tree 220. The actual tree nodes, as well as the record data, are not contained within the list 212, but are simply pointed to by the list members. Other implementations are possible, however.

The computer 202 builds a balanced binary tree by taking the linked list 212, successively subdividing it into smaller and smaller groupings, and linking the median of

each grouping with the median of the previous grouping. Specifically, the list is divided into a left side grouping and a right side grouping. The median of the list is then made the root of the binary tree. If the list has an even number of elements, then the left side median is used as the overall median. *ok it is clarified* The median element of the left side grouping is then determined using the same procedure. This second median element will become the left child of the root. The left side grouping is then divided into a left and right side groupings, with the median becoming the left child of the second median element. This procedure continues until there are no more left side groupings in the list.

The procedure then involves traveling back up the partly assembled binary tree to pick up any right side groupings. The right side groupings themselves are repeatedly subdivided into further groupings, with each median element linked to the previous median element as its right child in the binary tree. Whenever there is a choice between processing a left side grouping and a right side grouping, the left side grouping takes precedence. However, it is understood that this is only a matter of convention, and that the entire procedure can be altered so that the right side elements are chosen as the medians (in cases of even numbered groupings) and so that the right side groupings are always processed first. Each time a node is placed within the partly assembled binary tree, the Boolean flag 222 of the corresponding element in the linked list 212 is set to TRUE. This prevents an element from being placed more than once in a tree.

Referring to the flowchart of FIG. 5, with appropriate reference to FIGS. 6-11, an example of steps that may be followed by a computer to assemble a balanced binary tree will now be described. To help illustrate the steps, it will be assumed that the procedure is being performed using the setup of FIGS. 3 and 4, and appropriate reference will be made thereto.

At step 250, the computer 202 (FIG. 3) determines which element of the list 212 (FIG. 4) is the median element. Since there are an even number of elements, the object 5b (representing record 5a of FIG. 3) is treated as the overall median. The node 5c is then made the root of the binary tree, and its Boolean flag is set to TRUE, as shown in FIG. 6. At step 252, the computer 202 determines which element is the median of the left side grouping. The left side grouping includes those elements of the linked list that are to the left of 5b – namely, elements 1b, 2b, 3b and 4b. Again, since there are an even number of elements in this grouping, the object 2b is used. The node 2c is then made the left child of the node 5c, and its Boolean flag is set to TRUE, as shown in FIG. 7. The flow then continues to step 256, at which the computer 202 determines whether there are any unplaced elements to the left of object 2b. In this case, object 1b remains as a left side grouping of object 2b. The flow then moves to step 258, at which the object 2b is now treated as a parent object for the next steps. Steps 252 through 256 are then repeated, during which the object 1b is treated as the median of the left side grouping and the node 1c is placed as the left child of node 2c, as shown in FIG. 8. The flow then moves to step 260, since there are no more elements to the left of object 1b.

At step 260, the computer 202 moves back one level on the partially assembled binary tree to the previous node. At step 262, the computer checks to see if the element in the list 212 corresponding to the previous node has any elements to the right of it. In this case, the previous node was node 2c (FIG. 8), so the computer looks at object 2b in the list 212 and sees that 2b does, in fact, have elements to the right of it in its grouping. Specifically, objects 3b and 4b are to the right of object 2b in the left half of the list 212, with object 3b as the median of that small sub-grouping. The flow then continues to steps 264-268, in which the



node 2c is treated as a parent node and node 3c as its right child (FIG. 9). The computer then determines whether object 3b has any unplaced elements to the left of it at step 270. Since there are none, the flow proceeds to step 272, at which the computer looks at the right side of object 3b, and sees that there is one unplaced element in the same sub-grouping as object 3b, namely object 4b. Steps 262 and 268 are repeated, and node 4c is placed as the right child of node 3c (FIG. 10). The steps of FIG. 5 are repeated until the computer has walked back up the binary tree to the root node, at which point it determines that the tree is complete. The computer 202 then deletes the list 212, leaving only the completed, balanced tree as shown in FIG. 11.

There are many ways in which the above-described procedure may be implemented as a computer program. In an embodiment of the invention, the procedure is implemented as a function that receives a pointer to the linked list 212 (FIG. 4) and a pointer to the unassembled binary tree 220. The function then recursively divides the linked list, passes the median to itself, divides again, etc., until it can divide the list no more. The function then starts filling in the bottommost node on the left side of the tree and walks its way back up. As it makes its way back up, it fills in the remaining nodes. Referring to the flowchart of FIG. 12, an example of how such a function may work will now be described.

The term "present median" will be used to refer to the median of that has been passed to the instance of the function being used in a step, while the term "returned median" will be used to refer to a median that has been returned by another instance of the function used in another step. At step 320, the function locates the median element of the list 212 (FIG. 4), and establishes the corresponding node as the root of the tree. At step 322, the function



locates the median element of the left side group. At step 324, the function determines whether there are any more elements to the left of the present median. If there are, then the function calls itself again (step 340) and the new instance of the function moves back to step 322, using the elements to the left of the present median as the new left side grouping. If there are no more elements to the left side of the present median, then the flow proceeds to step 326, at which the function returns a pointer to the tree node corresponding to the present median to the instance of the function from which it was originally called, and the Boolean flag of the present median is set to TRUE. At step 330, the calling function links the tree node referenced by the returned median to the present median. The returned median is linked as either the left child or right child of the present median, depending on which side of the present median it is located in the linked list. At step 332, the function determines whether there are any elements to the right of the present median. If there are, the flow proceeds to step 333. If not, then the flow moves to step 328. If, at step 328, the chain of recursions is complete, then the procedure ends. Otherwise, it returns to step 324.

If, at step 332, the function determines that there are elements to the right of the present median, then the flow proceeds to step 333, at which the function calls itself and passes the present median to the new instance. In the new instance of the function (step 334), the function locates the median of the right side grouping of the present median. The flow then moves to step 336. At step 336, the function determines whether there are any elements to the left of the present element. If there are, then the flow moves to step 340. If not, then the flow moves to step 338, at which the function calls itself and passes the present median to the new instance.

Referring to FIG. 13, a more specific embodiment of the invention will now be described using elements introduced in FIG. 3. In this embodiment, the server computer 200 executes a multithreaded program 200a whose performance is being monitored at the client computer 202. Specifically, the program 200a is running several threads of execution  
5 through various code modules. Each time it passes through a different code module, it logs the event in a log 350. Once the program 200a has completed a series of test runs, the client computer 202 retrieves the log data and organizes it into a binary tree according to one or more of the procedures described above.

It can thus be seen that a new a useful method and system for creating a balanced  
10 binary tree has been provided. In view of the many possible embodiments to which the principles of this invention may be applied, it should be recognized that the embodiments described herein with respect to the drawing figures is meant to be illustrative only and should not be taken as limiting the scope of invention. For example, those of skill in the art will recognize that the elements of the illustrated embodiments shown in software may be  
15 implemented in hardware and vice versa or that the illustrated embodiments can be modified in arrangement and detail without departing from the spirit of the invention. Therefore, the invention as described herein contemplates all such embodiments as may come within the scope of the following claims and equivalents thereof.